**HTL Saalfelden**

**Systemplanung und Projektentwicklung**

****

**Projektdokumentation**

**2023 / 2024**

|  |  |
| --- | --- |
| **Projektbezeichnung** | Mbot-Steuerungssystem |
| **Projektteam** | David Legenjovic, Jonas Maier, Daniel Jessner |
| **Erstellt am** | 10.01.2024 |
| **Letzte Änderung am** | 17.04.2024 |
| **Status** | in Bearbeitung |
| **Aktuelle Version** | 1.4 |

**Änderungsverlauf**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Nr.** | **Datum** | **Version** | **Geänderte Kapitel** | **Art der Änderung** | **Autor** |
| 1 | 10.01.2024 | 1.0 | Alle | Erstellung | David Legenjovic |
| 2 | 21.03.2024 | 1.1 | Sprint 1, Sprint 2, Sprint3, Projektteam und Schnittstellen | Sprints beschrieben | David Legenjovic |
| 3 | 03.04.2024 | 1.2 | Nicht-Funktionale Anforderungen, Projektplanung | Nicht-Funktionale Änderungen hinzugefügt, Projektplanung Einleitung + Allgemeine Informationen hinzugefügt | Daniel Jessner |
| 4 | 10.03.2024 | 1.3 | Projektplanung, Funktionale Anforderungen | +: Variantenbildung, Machbarkeitsstudie, Projektziele, benötigte Ressourcen, Use Case 1 | Daniel Jessner |
| 5 | 17.03.2024 | 1.4 | Projektplanung, Entwicklungsmethodik, Kommunikation, Projektrisiko | +: Entwicklungsmethodik, Kommunikation, Projektrisiko | Daniel Jessner |
| 6 | 24.04.2024 | 1.5 | Projektplanung, Entwicklungsmethodik, Kommunikation, Projektrisiko | +: Projektumfeldanalyse | Daniel Jessner |

**Inhalt**

[1. Allgemeines / Projektübersicht 4](#_Toc965522712)

[1.1 Projektbeschreibung 4](#_Toc1040632592)

[1.2 Projektteam und Schnittstellen 4](#_Toc284604490)

[2. Funktionale Anforderungen 4](#_Toc521691458)

[2.1 Use Cases 4](#_Toc1840059643)

[2.1.1 <Name Use Case 1> 4](#_Toc1384425644)

[2.1.2 <Name Use Case 2> 4](#_Toc2131939576)

[2.1.3 <Name Use Case n> 4](#_Toc1332338971)

[3. Nichtfunktionale Anforderungen 4](#_Toc542409565)

[4. Projektplanung 5](#_Toc1181161558)

[4.1 Variantenbildung 5](#_Toc191553135)

[4.2 Machbarkeitsstudie 5](#_Toc729802831)

[4.3 Allgemeine Planungsinformationen 5](#_Toc2024584587)

[4.4 Projektumfeldanalyse 5](#_Toc502666407)

[5. Softwarearchitektur 5](#_Toc2079537197)

[5.1 Aktivitätsdiagramme 6](#_Toc659892040)

[5.1.1 Aktivitätsdiagramm 1 Name 6](#_Toc1701078734)

[5.1.2 Aktivitätsdiagramm n Name 6](#_Toc1952219604)

[5.2 Sequenzdiagramme 6](#_Toc452400050)

[5.2.1 Sequenzdiagramm 1 Name 6](#_Toc539048854)

[5.2.2 Sequenzdiagramm n Name 6](#_Toc1656117885)

[5.3 Komponentendiagramme 6](#_Toc1549670347)

[5.4 Verteilungsdiagramme 6](#_Toc1782933099)

[5.5 Softwarekomponenten / Programme 6](#_Toc863584749)

[5.5.1 SW Programme 6](#_Toc2059131748)

[5.5.2 SW Komponenten 6](#_Toc2020768912)

[6. Projektdurchführung 7](#_Toc1459545787)

[6.1 Sprint 1 7](#_Toc965912477)

[6.1.1 Sprintplanung 7](#_Toc1238068464)

[6.1.2 Sprint Demo 7](#_Toc1608356264)

[6.1.3 Sprint Retrospektive 7](#_Toc1207657317)

[6.1.4 Sprint Zusammenfassung 7](#_Toc56107529)

[6.2 Sprint 2 8](#_Toc501505208)

[6.2.1 Sprintplanung 8](#_Toc1492261807)

[6.2.2 Sprint Demo 8](#_Toc577798987)

[6.2.3 Sprint Retrospektive 8](#_Toc194081203)

[6.2.4 Sprint Zusammenfassung 8](#_Toc729203803)

[6.3 Sprint n 8](#_Toc562254915)

[7. Installation / Software deployment 8](#_Toc1526420175)

[8. Projektabschluß 8](#_Toc1271613368)

[8.1 Projektzusammenfassung 8](#_Toc1743416473)

[8.2 Attachments 8](#_Toc1717973310)

! -> NICHT ERLEDIGT

# Allgemeines / Projektübersicht

## Projektbeschreibung

Dieses Projekt ist ein Semesterprojekt der 4. AHINF der HTL Saalfelden und hat als Ziel, ein System zu entwickeln, welches einen sogenannten „mBot2“ des Unternehmens „makeblock" fernsteuert. Dazu eignet sich eine Dreiteilung in der Entwicklung in die Bereiche Server, Steuerungssoftware und den Mbot an sich. Basisanforderungen des Projektes geben vor, dass die GUI, die dem Benutzer zur Verfügung gestellt wird, sowohl am Windows-System als auch auf mobilen Endgeräten laufen soll. Weiters wird der gesamte Projektverlauf unten dokumentiert und jegliche Änderungen mit Version Control via „git“ festgehalten.

## Projektteam und Schnittstellen

|  |  |  |  |
| --- | --- | --- | --- |
| **Rolle(n)** | **Name** | **Telefon** | **E-Mail** |
| Client-Entwickler | David Legenjovic | +43 677 61331252 | [david.legenjovic@htl-saalfelden.at](mailto:david.legenjovic@htl-saalfelden.at) |
| Server-Entwickler | Jonas Maier | +43 664 75087002 | [jonas.maier@htl-saalfelden.at](mailto:jonas.maier@htl-saalfelden.at) |
| mBot-Entwickler | Daniel Jessner | +43 650 6349636 | [daniel.jessner@htl-saalfelden.at](mailto:daniel.jessner@htl-saalfelden.at) |

# Funktionale Anforderungen !

## Use Cases !

### MBot mit dem Internet und Server verbinden

* **Akteure:**

Testperson (als Benutzer), MBot (als Client), (Schul-)Netzwerk (als Zugang zum Internet), Server (als Vermittler zwischen MBot und Steuerungssoftware

* **Auslöser / Trigger-Event:**

Einschalten des MBots

* **Kurzbeschreibung:**

Testperson testet den Boot-Vorgang des MBots, welcher sich automatisch zum Internet und zum Server verbindet.

* **Beschreibung der einzelnen Schritte:**

Testperson schaltet, nachdem sie die korrekten Konfigurationen eingetragen hat, den MBot ein, welcher sich daraufhin automatisch mit dem angegeben W-LAN verbindet und dann eine Kommunikation zum Server aufbaut.

* **Beschreibung alternativer Schritte:**

Die Testperson trägt versehentlich inkorrekte Konfigurationen ein (Server oder W-LAN) und ermöglicht es dem MBot so nicht erfolgreiche Verbindungen herzustellen. Dieser erkennt, dass keine Verbindung besteht und versucht in einer Endlosschleife, sich erneut zu verbinden.

* **Vor- und Nachbedingungen**:

Es soll möglich sein, einfach und intuitiv den MBot erfolgreich zu starten.

* **Systemgrenzen- und Fehler:**

Keine

Use-Case-Diagramm:

### Steuerungssoftware starten und mit Server verbinden

### 2.1.3 <Name Use Case n>

# Nichtfunktionale Anforderungen

## Benutzerfreundlichkeit

Die GUI der Steuerungssoftware sollte intuitiv gestaltet sein, um eine einfache Bedienung für Benutzer aller Erfahrungsstufen zu gewährleisten. Außerdem sollte die Reaktionszeit der Steuerungssoftware sowie die Latenz zwischen Software und Mbot so gering wie möglich sein, um ein reibungsloses und ansprechendes Benutzererlebnis zu ermöglichen.

## Zuverlässigkeit

Es ist ein wichtiger Punkt, dass die Verbindung zwischen Mbot und der Software, also auch über den Server, stabil ist, damit eine konstante, ununterbrochene Steuerung ohne Probleme zur Verfügung gestellt werden kann. Weiters soll das System so weit robust sein, dass es angemessen auf Ausnahmesituationen reagiert, um einen kontinuierlichen Betrieb auch bei unvorhergesehenen Ereignissen sicherzustellen.

## Skalierbarkeit

Es ist gewünscht, das System so zu gestalten, dass es auch in der Zukunft liegende Erweiterungen und zusätzliche Änderung ohne großen Aufwand unterstützt. Die Software sollte die Funktionalität haben, dem Benutzer mehrere gefundene Mbots zur Auswahl zu stellen, ohne dabei die Perfomance oder Stabilität des Systems zu beeinträchtigen.

## 3.4 Sicherheit

Die Kommunikation zwischen Mbot, Server und Software sollte sicher und privat sein, um äußeren Einfluss beziehungsweise Manipulation zu verhindern.

## 3.5 Plattformkompatibilität

Wie auch die Grundanforderungen des Projektes vorgeben, sollte die Software derartig unabhängig sein, dass die Mbot-Steuerung sowohl in einer Windows-Umgebung als auch auf mobilen Endgeräten (Android) funktioniert. Dabei ist die Responsivität der Software von großer Wichtigkeit, um eine konsistente Darstellung zu jeder Zeit zu gewährleisten.

## 3.6 Dokumentation und Wartbarkeit

Das Projekt sollte ausführlich dokumentiert sein, einschließlich Anleitungen zur Installation, Konfiguration und Verwendung der Steuerungssoftware. Dazu gehört auch eine gute Strukturierung sowie Kommentierung des Quellcodes, um eine einfache Wartung und Weiterentwicklung des Systems zu ermöglichen.

# Projektplanung !

Die Projektplanung ist ein entscheidender Schritt bei der Durchführung eines jeden Projekts, da sie den Rahmen setzt, innerhalb dessen das Projekt durchgeführt wird, und die Grundlage für einen erfolgreichen Abschluss bildet. Dieser Prozess ermöglicht es, die Ziele, Anforderungen, Ressourcen und Zeitpläne zu definieren und zu organisieren, um sicherzustellen, dass das Projekt effizient und effektiv umgesetzt wird.

Für das vorliegende Projekt ist eine gründliche Projektplanung von entscheidender Bedeutung. Das Ziel dieser Planung ist es, die Schritte und Maßnahmen festzulegen, die erforderlich sind, um das Projekt erfolgreich abzuschließen, sowie potenzielle Risiken zu identifizieren und Strategien zu entwickeln, um ihnen zu begegnen.

Die Projektplanung wird sich auf verschiedene Aspekte konzentrieren, darunter die Definition der Projektziele, die Bestimmung der benötigten Ressourcen, die Festlegung eines Zeitplans und die Identifizierung von Verantwortlichkeiten und Zuständigkeiten innerhalb des Projektteams. Darüber hinaus wird die Planung auch die Auswahl der geeigneten Entwicklungsmethodik, die Kommunikations- und Berichterstattungsstrategien sowie die Risikobewertung und -management umfassen.

## Variantenbildung

* Mögliche Varianten:
  + Dreiteilung (mBot, Software, Server)
  + Zweiteilung (mBot, Software+Server)
  + Mit Datenbank, ohne Datenbank
  + Nosql, Mysql

Schlussendlich wurde, wie schon kurz erwähnt, die Dreiteilung mit Datenspeicherung in einer Mongo-Datenbank gewählt, da dieser Weg es ermöglicht, sich jeweils voll und ganz auf einen Bereich zu konzentrieren. Der mBot kümmert sich um die Verarbeitung der Befehle, der Server um die Weiterleitung (Schnittstelle zwischen Software und mBot) und die Steuerungssoftware um die Bedienung durch den Benutzer.

Weiters hat sich das Projektteam für eine einfache, unkomplizierte und schnelle Datenspeicherung für die Nosql-Variante mittels MongoDB entschieden, um das gespeicherte Datenmodell möglichst flexible zu halten und nicht erforderlichen Aufwand einer SQL-Datenbank bezüglich Beziehungen, etc. zu vermeiden.

## Machbarkeitsstudie

Mittels Übungsstunden vor dem Projektstart wurde dem Projektteam der mBot und die mBlock-Umgebung nähergebracht und erste Versuche damit angestellt. Es kristallisierten sich zwei Teilbereiche der Machbarkeitsanalyse also schon im Vorhinein heraus:

* **Technische Machbarkeit:**
  + Die erforderliche Hardware (mBot-Roboter, Sensoren) ist kommerziell erhältlich und relativ kostengünstig.
  + Programmierumgebungen wie Scratch und Arduino IDE oder die konzerneigenen „mBlock“-Umgebung bieten eine benutzerfreundliche Plattform für die Entwicklung und Programmierung des Roboters.
  + Die grundlegenden Funktionen (Hindernisvermeidung, Linienverfolgung) sind technisch machbar und wurden in ähnlichen Projekten bereits erfolgreich umgesetzt.
  + Der Microcontroller des mBot ist mit Micropython kompatibel und lässt einen gewissen Funktionsbereich von Python zu.
* **Zeitliche Machbarkeit:**
  + Die Montage des mBot-Roboters erfordert in der Regel weniger als eine Stunde (in diesem Projekt nicht von Entwickelnden durchgeführt).
  + Die Programmierung der grundlegenden Funktionen kann je nach Kenntnisstand und Erfahrung des Entwicklers einige Stunden bis Tage in Anspruch nehmen.
  + Testen und Feinabstimmung des Roboters erfordern zusätzliche Zeit, um sicherzustellen, dass er zuverlässig funktioniert.
  + Trotz oben angeführten Punkten ist das Projektteam davon überzeugt, dass genügend Zeit zur Verfügung steht

Grundsätzlich ist das Projekt in der Machbarkeit als sehr gut zu bewerten, da im Vorhinein keine besorgniserrengenden oder möglich problematische Aspekte erkannt werden können.

## Allgemeine Planungsinformationen

Hier werden andere Planungsinformationen angeführt, welche nicht durch andere Kapitel abgedeckt sind, jedoch durchaus für das vorliegende Projekt von Bedeutung sind.

### Projektziele

**Grundlegende Steuerungsfunktionen:**

Implementierung von Basissteuerungsfunktionen wie Vorwärtsfahren, Rückwärtsfahren, Links- und Rechtsdrehungen, um eine grundlegende Navigation des mBot-Roboters zu ermöglichen (hier durch einen Joy Stick geregelt).

**Hindernisvermeidung:**

Entwicklung eines Systems zur Hindernisvermeidung mithilfe von Ultraschallsensoren, um den mBot autonom navigieren zu lassen und Zusammenstöße mit Hindernissen zu vermeiden.

**Linienverfolgung:**

Integration eines Linienverfolgungssystems, das es dem mBot ermöglicht, einer Linie auf dem Boden zu folgen und sich entlang vorgegebener Pfade zu bewegen. Dieser Modus ist partikulär durch Basisanforderungen an die Steuerungssoftware gebunden.

**Benutzerinteraktion:**

Implementierung einer Benutzerschnittstelle zur Steuerung des mBot über die Steuerungssoftware. Diese Schnittstelle wird in diesem Projekt als Joy Stick realisiert.

**Modularität und Erweiterbarkeit:**

Design des Steuerungssystems mit einer modularen Architektur, die es ermöglicht, neue Sensoren, Aktuatoren oder Steuerungsfunktionen einfach hinzuzufügen oder zu entfernen, um die Funktionalität des mBot zu erweitern.

**Robuste Leistung:**

Gewährleistung einer robusten Leistung des Steuerungssystems unter verschiedenen Umgebungsbedingungen, einschließlich unterschiedlicher Bodenbeschaffenheiten, Lichtverhältnisse und Hinderniskonfigurationen, so weit wie die Hardware es zulässt.

**Einfachheit und Benutzerfreundlichkeit:**

Entwicklung eines benutzerfreundlichen Steuerungssystems mit einer intuitiven Benutzeroberfläche und klaren Anweisungen, um die Konfiguration und Interaktion mit dem mBot für Benutzer aller Erfahrungsstufen zugänglich zu machen.

**Dokumentation:**

Bereitstellung umfassender Dokumentation, einschließlich Anleitungen, um Benutzern bei der Installation, Konfiguration und Nutzung des Steuerungssystems zu unterstützen.

**Sicherheit und Zuverlässigkeit:**

Priorisierung von Sicherheit und Zuverlässigkeit bei der Entwicklung des Steuerungssystems, um sicherzustellen, dass der mBot ordnungsgemäß funktioniert und potenzielle Risiken oder Gefahren minimiert werden.

### Benötigte Ressourcen

**mBot-Roboter:**

Der mBot-Roboter ist die zentrale Hardwarekomponente des Projekts. Er besteht aus Motoren, Rädern, einem Mikrocontroller, Sensoren und anderen elektronischen Komponenten.

**Sensoren:**

Die Sensoren des mBot sind Bauteile die am Gerät selbst, welche dem System interessante Daten bereitstellen, welche in der Datenbank abgespeichert werden und dem Benutzer an GUI anschaulich präsentiert werden. Diese können modular ergänzt werden. Sensoren, welche in diesem Projekt zum Einsatz kommen, sind:

* Ultraschallsensor:

Zur Hindernisvermeidung und zur Messung von Entfernungen.

* Lichtsensor:

Für Lichtmessungen oder zur Erkennung von Umgebungslichtbedingungen.

* Front-Light-Sensors:

Insgesamt vier Stück zur Messung der Untergrundfarbe. Wird für Linienverfolgungsmodus verwendet.

* Lage-Sensoren (pitch, yaw, roll):

Insgesamt drei Sensoren, welche Daten über Neigung, Drehung um x-Achse und Drehung um y-Achse liefern. Diese helfen dabei, die Lage des mBot visuell in der Steuerungssoftware darzustellen.

* Geräuschsensor:

Für die Messung von Umgebungsgeräuschen.

* Erschütterungssensor:

Misst das Maß an Erschütterung, das den mBot erreicht. Je größer der gemessene Wert, desto stärker wird der mBot durch Erschütterung beeinflusst („gewackelt“).

**Programmierbare Hardwareplattform:**

Für dieses Projekt fiel die Wahl auf den mBot. Dieser bietet mit dem CyberPi eine programmierbare Hardwareplattform, die eine benutzerfreundliche Programmierung des mBots ermöglicht und mit verschiedenen Sensoren und Aktuatoren kompatibel ist.

Anderweitig wäre auch noch „mCore“ als Alternative dazu existent, ist für dieses Projekt aber nicht von Relevanz.

**Programmiersoftware:**

Neben der berücksichtigten Arduino IDE, welche eine populäre Entwicklungsumgebung für die Programmierung des mBot in der Arduino-Sprache bereitstellt, hat sich das Projektteam für die vom mBot-Hersteller veröffentliche Umgebung „mBlock“ entschieden, da diese mit dem integrierten Python-Editor eine einfache und schnelle Möglichkeit bietet, Programme für den mBot in Micro Python zu entwickeln.

**Zusätzliche Bauteile:**

Obwohl das Projektteam an sich nicht am Zusammenbau des mBot beteiligt war, sind durchaus die einzelnen Komponenten am Gerät zu erkennen:

* **Batterien**:

Zur Stromversorgung des mBot-Roboters und anderer Komponenten.

* **Verbindungskabel**: Zum Anschließen von Sensoren, Aktuatoren und anderen elektronischen Bauteilen, aber auch für die Verbindung zur mBlock-DIE.
* **Gehäuse oder Rahmen**: Optional für die Montage und den Schutz des mBot-Roboters und seiner Komponenten.

**Dokumentation und Anleitungen:**

Während des Projektzeitraumes hat sich das Entwicklerteam regelmäßig externen Input geholt, um Probleme zu lösen oder einfach Informationen über gewisse Funktionalitäten zu bekommen:

* Handbücher:

Zu den mBot-Roboterbausätzen und den einzelnen Sensoren (auch direkt von mBlock zur Verfügung gestellt)

* Online-Tutorials:

Anleitungen und Informationen nicht nur über die Programmierung des mBot, sondern auch zu Avalonia (Software) und Springboot (Server)

* Community-Foren:

Online-Plattformen, auf denen Benutzer Fragen stellen, Erfahrungen austauschen und Unterstützung erhalten können (bsp.: [stackoverflow](http://www.stackoverflow.com)).

**Zeit und Engagement:**

Das Projektteam berechnet Zeit sowohl für die Programmierung des Steuerungssystems und die Durchführung von Tests und Feinabstimmungen als auch für das Engagement für kontinuierliches Lernen und Experimentieren mit neuen Funktionen und Technologien.

Berücksichtigt man die 3 Übungsstunden pro Woche, die für das Projekt vorgesehen sind, ergeben sich also grundsätzlich 6h/Sprint.

### 4.3.3 Entwicklungsmethodik

Die Entwicklungsmethodik bezieht sich auf die Wahl des Ansatzes, der nicht nur für die Durchführung, sondern auch für die Planung des Projektes verwendet wird. Dieses Projekt wird anhand der agilen Projektmanagement-Methode „**Scrum**“ geplant, geleitet und bewertet.

Das bedeutet: Anforderungen werden auf User Stories heruntergebrochen, geschätzt (in Value Points & Story Points) und bei den jeweiligen Sprintplanungen (alle 2 Wochen) vom Product Backlog in das Sprint Backlog übernommen. Am Ende eines Sprints findet ein Sprint Review und (optional) eine Sprint Retrospektive.

Ausschlaggebend bei dieser Methode ist der kontinuierliche Kontakt zum Kunden. In diesem Fall wird am Ende eines Sprints der Projektfortschritt den leitenden Lehrpersonen präsentiert und sozusagen direktes Feedback von den „Kunden“ eingeholt.

### 4.3.4 Kommunikations- und Berichterstattungsstrategie

Wie schon im obigen Punkt „Entwicklungsmethodik“ erwähnt wurde, werden alle 2 Wochen alle Projektfortschritte den Stakeholdern präsentiert. In diesen Präsentationen wird darauf eingegangen, welche Sprintziele erreicht wurden und welche nicht, welche User Stories im Sprint Backlog verbleiben und an den nächsten Sprint weitergegeben werden und welche erledigt sind und auch wie viele Story Points insgesammt noch im Product Backlog verbleiben. Zusätzlich werden der Projektverlauf und die Sprint Velocity („Schnelligkeit des Entwicklungsteams“) in einem Burndown-Chart und einem Säulendiagramm dargestellt.

**Live-Demo:**

Um den Stakeholdern auch effektiv vor Augen zu führen, was sich im Projekt im Laufe des Sprints getan hat, wird eine Live-Demo zur Verfügung gestellt. Auf diese Weise können die Stakeholder den Fortschritt des Projekts auch auf persönlicher Ebene bewerten und Feedback abgeben.

**Dokumentation:**

Das ganze Projekt wird laufend in einer Dokumentation festgehalten, welche mit jedem Sprint aktualisiert wird. Diese könnte während, oder auch am Ende des Projektes bei möglich auftretenden Fragen helfen und den Stakeholdern eventuell einen besseren Einblick gewähren.

### 4.3.5 Projektrisiko(-bewertung)

Da das hier angeführte Projekt ein Schul-Projekt darstellt und in keinster Art und Weise kommerzielle oder finanzielle Auswirkungen mitsichzieht, ist ein solches Projektrisiko eher milder zu bewerten. Trotzdessen gibt es auch hier einen Punkt, der durchaus zu berücksichtigen ist:

**Mangelnde Verfügbarkeit von Ressourcen:**

Beschreibung:

Das Risiko besteht darin, dass möglicherweise nicht genügend Ressourcen wie Zeit, Materialien oder „Personal“ für das Schulprojekt zur Verfügung stehen.

Ursachen:

Dieses Risiko kann durch verschiedene Faktoren verursacht werden, darunter unvorhergesehene Verzögerungen oder unerwartete Änderungen in den Projektanforderungen.

(\*Projektanforderungen wurden 1x durch Stakeholder ergänzt)

Auswirkungen:

Wenn nicht genügend Ressourcen vorhanden sind, könnte dies zu Verzögerungen bei der Durchführung des Projekts führen, die Qualität des Endprodukts beeinträchtigen oder sogar dazu führen, dass das Projekt nicht rechtzeitig abgeschlossen wird.

Risikobewertung:

Die Wahrscheinlichkeit dieses Risikos hängt von verschiedenen Faktoren ab, darunter die Komplexität des Projekts, die Verfügbarkeit von Ressourcen und die Erfahrung des Projektteams.

Risikominderungsstrategien:

Um das Risiko mangelnder Ressourcen zu mindern, könnten Maßnahmen ergriffen werden wie die frühzeitige Identifizierung und Planung von benötigten Ressourcen, die Aufstellung eines realistischen Zeitplans, die Suche nach zusätzlichen Finanzierungsquellen oder die Einbindung zusätzlicher Unterstützungspersonen, Sprichwort: **Scrum**.

Verantwortlichkeiten:

Die Projektverantwortlichen sind dafür verantwortlich, das Risiko zu überwachen, frühzeitig auf Anzeichen von Ressourcenmangel zu reagieren und geeignete Maßnahmen zur Risikominderung zu ergreifen.

## Projektumfeldanalyse

**Zielsetzung des Projekts:**

Das Hauptziel des Projekts ist die Entwicklung eines Systems zur Fernsteuerung des mBot2 von Makeblock. Die Aufgabe ist in drei Hauptbereiche unterteilt: Server, Steuerungssoftware und mBot2.

**Anforderungen an das Projekt:**

* Das System soll eine grafische Benutzeroberfläche (GUI) haben, die sowohl auf Windows-Systemen als auch auf mobilen Endgeräten funktioniert.
* Es müssen Basisfunktionalitäten implementiert werden, um die Kommunikation zwischen der Steuerungssoftware und dem mBot2 zu ermöglichen.
* Der gesamte Projektverlauf muss dokumentiert werden, und Änderungen sollen mit Hilfe von Version Control via Git festgehalten werden.

**Technologische Anforderungen:**

* Entwicklung von GUI-Software für Windows und mobile Endgeräte.
* Implementierung einer Serverkomponente zur Verbindung zwischen Steuerungssoftware und mBot2.
* Programmierung des mBot2, um Befehle von der Steuerungssoftware entgegenzunehmen und auszuführen.

**Rahmenbedingungen:**

* Das Projekt wird im Rahmen eines Semesterprojekts der 4. AHINF der HTL Saalfelden durchgeführt.
* Es gibt Zeitbeschränkungen, die mit dem Ende des Semesters einhergehen.
* Ressourcen wie Unterrichtsraum, Laborausrüstung (mBot2, Sensoren, Netzwerk, …) und finanzielle Unterstützung (hauptsächlich mBot2 und Sensoren) stehen durch die Schule zur Verfügung.
* Expertise der der beteiligten Schüler

**Externe Faktoren:**

* Verfügbarkeit von Hardware (mBot2, Computer, mobile Geräte) und Software (Entwicklungsumgebungen, Bibliotheken).
* Eventuelle externe Hilfe muss berücksichtigt werden. Dazu gehören Frage-Foren wie z.B. die bereits erwähnte Seite „stackoverflow“.
* Risiken wie unvorhergesehene technische Schwierigkeiten, begrenzte Ressourcen oder Zeitmangel sollten berücksichtigt werden.

**Wettbewerbsfaktoren:**

* Erfolgreiche Projekte aus früheren Semestern oder ähnliche Projekte an anderen Bildungseinrichtungen könnten als Benchmark dienen.
* Innovationspotenzial: Gibt es Möglichkeiten, das Projekt von anderen ähnlichen Projekten abzuheben? -> mBot2 als Mittelpunkt

**Projektmanagement:**

* Eine klare Aufgabenverteilung und Zeitplanung sind entscheidend, um sicherzustellen, dass das Projekt rechtzeitig abgeschlossen wird.
* Kommunikation und Zusammenarbeit zwischen den Teammitgliedern sind unerlässlich, insbesondere bei einem Projekt mit verschiedenen technischen Aspekten wie diesem.

**Rechtliche Aspekte:**

* Es ist wichtig sicherzustellen, dass alle verwendeten Technologien und Softwarelizenzen den rechtlichen Anforderungen entsprechen.
* Datenschutz- und Sicherheitsaspekte müssen ebenfalls berücksichtigt werden, insbesondere wenn das System drahtlos kommuniziert und persönliche Daten verarbeiten könnte (Datenspeicherung in Datenbank -> eventuelle Login-Erweiterung)
* Diese Analyse bietet einen umfassenden Überblick über das Projektumfeld und dient als Leitfaden für die Planung, Durchführung und Bewertung des Projekts zur Entwicklung des mBot2-Fernsteuerungssystems.

# Softwarearchitektur !

In diesem Kapitel soll der Aufbau der Software/Hardware beschrieben werden.

aus welchen Komponenten besteht das SW-Produkt.

Wie interagieren die einzelnen Komponenten miteinander.

Auf welcher Hardware läuft das System bzw. handelt es sich vielleicht sogar um ein verteiltes System. Wie kommunizieren diese Komponenten miteinander.

## Aktivitätsdiagramme !

### Aktivitätsdiagramm 1 Name

Beschreibung der entsprechenden Aktivität und einfügen des Aktivitätsdiagramm

### Aktivitätsdiagramm n Name

Beschreibung der entsprechenden Aktivität und einfügen des Aktivitätsdiagramm

## Sequenzdiagramme !

### Sequenzdiagramm 1 Name

Beschreibung der entsprechenden Sequenz und Einfügen des Sequenzdiagramm

### Sequenzdiagramm n Name

Beschreibung der entsprechenden Sequenz und Einfügen des Sequenzdiagramm

## Komponentendiagramme !

Aus welchen Komponenten besteht die Software und welche Schnittstellen bieten diese an.

## Verteilungsdiagramme !

Zeigt an wie die einzelnen Teile der Software auf die Hardwarekomponenten verteilt sind und wie die Hardwarekomponenten miteinander verbunden sind.

Auf welchem Rechner läuft welcher Software. Wie sind diese über ein Netzwerk miteinander verbunden.

## Softwarekomponenten / Programme

### SW Programme

Dieses Projekt wurde in unterschiedlichen Programmen bearbeitet und bediente sich selbst auch welcher, z.B. für Dokumentation oder Absicherung des Fortschrittes.

Folgende **Programme** wurden verwendet:

|  |  |  |  |
| --- | --- | --- | --- |
| **Programm** | **Version** | **Hersteller** | **Beschreibung** |
| Visual Studio | 2022 (17.9.1) | Microsoft | Entwicklungsumgebung für C# |
| IntelliJ IDEA | 2023 (2.6) | JetBrains | Entwicklungsumgebung für Java |
| mBlock | 5.4.3 | makeBlock | Entwicklungsumgebung für |
| Github Desktop | 3.3.13 (x64) | Github, Inc. | Versionskontroll-Software zur Code-Sicherung |

### SW Komponenten

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **SW-Komponent** | **Version** | **Hersteller** | **Bezugsquelle** | **SW-Lizenz** | **Beschreibung** |
| Avalonia | 11.0.10 | AvaloniaUI OÜ | <https://www.nuget.org/packages/Avalonia> | <https://licenses.nuget.org/MIT> | Framework für Crossplattform-Entwicklung in C# (WPF) |
|  |  |  |  |  |  |

Auflistung aller verwendeten SW Komponenten welche für den Betrieb der SW benötigt werden. z.B.: Java Version, Apache Webserver, DotNet Framework, SW Library XY

inkl. Versiosnummer, Hersteller, Bezugsquelle (Downloadlink, ...) und SW-Lizenz (GPL, LGPL, Apache License, ...

# Projektdurchführung !

## Sprint 1

### Sprintplanung

Dauer: 07.02.2024-06.03.2024

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Nummer | Kurzbeschreibung | Story Points |
| Create Software | 3 | Create Software to Monitor and Controll MBots | 1 |
| Connect Mbot to network | 8 | Make MBot connect to | 3 |
| Setup Spring Server | 15 | Setup the Server which connects MBots and client | 21 |
| Get all MBots from Server | 6 | Client should receive all MBot Data | 5 |
| MBot receive input | 9 | Mbot should receive input from server to control its movement | 3 |
| Send user input to Mbot | 13 | Send the user input to the server which forwards it to the MBot | 5 |
| Make DetailedView | 7 | Make a detailed view of the mbot in the client | 5 |

Anzahl Story points: 43

Ausgewählte Punkte aus der Impediment Liste: noch keine Impediment-Liste vorhanden

### Sprint Demo

Kurze Beschreibung welche User Stories umgesetzt worden sind und welche Funktionen erfolgreich präsentiert worden sind.

Auflistung welche Punkte nicht umgesetzt werden konnten und warum.

|  |  |  |
| --- | --- | --- |
| Name | Nummer | Nicht, weil… |
| Create Software | 3 |  |
| Connect Mbot to network | 8 |  |
| Get all MBots from Server | 6 |  |
| Setup Spring Server | 15 | Zeitlich überschätzt |
| MBot receive input | 9 | Zeitlich überschätzt |
| Send user input to Mbot | 13 | Zeitlich überschätzt |
| Make DetailedView | 7 | Zeitlich überschätzt |

### Sprint Retrospektive

|  |  |
| --- | --- |
| Gut | Schlecht |
|  | Keine Branches verwendet |
|  | Kommunikationsschwierigkeiten |
|  |  |
|  |  |

|  |
| --- |
| Impediment Liste |
| Kommunikationsschwierigkeiten |
| Designfehler |
|  |
|  |
|  |
|  |
|  |

### Sprint Zusammenfassung

Wurden in diesem Sprint neue User Stories in das Product Backlog eingefügt und wenn ja, welche.

Wurden in diesem Sprint User Stories aus dem Product Backlog entfernt und wenn ja, welche und warum.

**Burndownchart**

**Sprint Velocity**

Durchschnittliche Velocity: 9

## Sprint 2

### Sprintplanung

Dauer: 06.03.2024-20.03.2024

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Nummer | Kurzbeschreibung | Story Points |
| Server Broadcast | 19 | Add code which listens to broadcasts | 5 |
| MBot send sensor data to server | 17 | Sends the data of the MBot to the server | 5 |
| Make DetailedView | 7 | Make a detailed view of the mbot in the client | 5 |
| Setup Spring Server | 15 | Setup the Server which connects MBots and client | 21 |
| MBot receive input | 9 | Mbot should receive input from server to control its movement | 3 |
| Send client commands to Mbot | 21 | Forwards the clients commands to the mbot | 5 |
| Send data to client | 23 | Send the MBots data continuously to the client | 8 |
| Send user input to the server | 13 | Send the commands to the server | 5 |

Anzahl Story points: 57

Ausgewählte Punkte aus der Impediment Liste: ---

### Sprint Demo

Kurze Beschreibung welche User Stories umgesetzt worden sind und welche Funktionen erfolgreich präsentiert worden sind.

Auflistung welche Punkte nicht umgesetzt werden konnten und warum.

|  |  |  |
| --- | --- | --- |
| Name | Nummer | Nicht, weil… |
| Server Broadcast | 19 |  |
| MBot send sensor data to server | 17 |  |
| Make DetailedView | 7 |  |
| Setup Spring Server | 15 |  |
| MBot receive input | 9 |  |
| Send client commands to Mbot | 21 | Zeitlich überschätzt |
| Send data to client | 23 | Zeitlich überschätzt |
| Send user input to the server | 13 | Zeitlich überschätzt |

### Sprint Retrospektive

|  |  |
| --- | --- |
| Gut | Schlecht |
|  | Kommunikationsschwierigkeiten |
|  |  |
|  |  |

|  |
| --- |
| Impediment Liste |
| Kommunikationsschwierigkeiten |
| Verständnisprobleme zur SW-Architektur |
|  |
|  |
|  |
|  |
|  |

### Sprint Zusammenfassung

Wurden in diesem Sprint neue User Stories in das Product Backlog eingefügt und wenn ja, welche.

Wurden in diesem Sprint User Stories aus dem Product Backlog entfernt und wenn ja, welche und warum.

**Burndownchart**

**Sprint Velocity**

Durchschnittliche Velocity: 20

## Sprint 3

### Sprintplanung

Dauer: 21.03.2024-10.04.2024

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Nummer | Kurzbeschreibung | Story Points |
| Server Broadcast | 19 | Add code which listens to broadcasts | 5 |
| MBot send sensor data to server | 17 | Sends the data of the MBot to the server | 5 |
| Make DetailedView | 7 | Make a detailed view of the mbot in the client | 5 |
| Setup Spring Server | 15 | Setup the Server which connects MBots and client | 21 |
| MBot receive input | 9 | Mbot should receive input from server to control its movement | 3 |
| Send client commands to Mbot | 21 | Forwards the clients commands to the mbot | 5 |
| Send data to client | 23 | Send the MBots data continuously to the client | 8 |
| Send user input to the server | 13 | Send the commands to the server | 5 |

Anzahl Story points: 57

Ausgewählte Punkte aus der Impediment Liste: ---

### Sprint Demo

Kurze Beschreibung welche User Stories umgesetzt worden sind und welche Funktionen erfolgreich präsentiert worden sind.

Auflistung welche Punkte nicht umgesetzt werden konnten und warum.

|  |  |  |
| --- | --- | --- |
| Name | Nummer | Nicht, weil… |
| Server Broadcast | 19 |  |
| MBot send sensor data to server | 17 |  |
| Make DetailedView | 7 |  |
| Setup Spring Server | 15 |  |
| MBot receive input | 9 |  |
| Send client commands to Mbot | 21 | Zeitlich überschätzt |
| Send data to client | 23 | Zeitlich überschätzt |
| Send user input to the server | 13 | Zeitlich überschätzt |

### Sprint Retrospektive

|  |  |
| --- | --- |
| Gut | Schlecht |
|  | Kommunikationsschwierigkeiten |
|  |  |
|  |  |

|  |
| --- |
| Impediment Liste |
| Kommunikationsschwierigkeiten |
| Verständnisprobleme zur SW-Architektur |
|  |
|  |
|  |
|  |
|  |

### Sprint Zusammenfassung

Wurden in diesem Sprint neue User Stories in das Product Backlog eingefügt und wenn ja, welche.

Wurden in diesem Sprint User Stories aus dem Product Backlog entfernt und wenn ja, welche und warum.

**Burndownchart**

**Sprint Velocity**

Durchschnittliche Velocity: 20

# Installation / Software deployment !

Folgende Schritte sind für die Installation der Steuerungs-Software notwendig:

Anleitung welche Schritte notwendig sind um das fertige SW Produkt zu installieren und in Betrieb zu nehmen.

# Projektabschluss !

## Projektzusammenfassung

Zusammenfassung der Projektdurchführung. Was lief gut/schlecht. Welche Erkenntnisse wurden während der Durchführung des Projektes gewonnen. Was würde man, nun anders machen bzw. wieder gleich machen?

## Attachments

Tabellarische Auflistung der Projektdateien.

z.B.: ZIP-File mit dem Quellcode, Projektpräsentationen, ...